Número Aleatorio

Un número aleatorio es un resultado de una variable al azar especificada por una función de distribución. Cuando no se especifica ninguna distribución, se presupone que se utiliza la distribución uniforme continua en el intervalo [0,1]

Los números aleatorios permiten a los modelos matemáticos representar la realidad.
En general cuando se requiere una **impredecibilidad** en unos determinados datos, se utilizan números aleatorios
Los seres humanos vivimos en un medio aleatorio y nuestro comportamiento lo es también. Si deseamos predecir el comportamiento de un material, de un fenómeno climatológico o de un grupo humano podemos inferir a partir de datos estadísticos. Para lograr una mejor aproximación a la realidad nuestra herramienta predictiva debe funcionar de manera similar: **aleatoriamente**. De esa necesidad surgieron los modelos de simulación.
En la vida cotidiana se utilizan números aleatorios en situaciones tan dispares como pueden ser los juegos de azar, en el diseño de la caída de los copos de nieve, en una animación por ordenador, en tests para localización de errores en chips, en la transmisión de datos desde un satélite o en las finanzas.

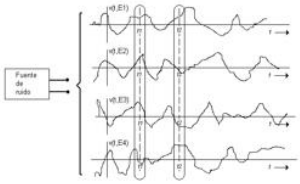
Como se pueden generar los números aleatorios

Para generarlos podemos aprovecharnos de situaciones reales para obtener una tabla de números aleatorios, como la lista de los números de Lotería Nacional premiados a lo largo de su historia, pues se caracterizan por que cada dígito tiene la misma probabilidad de ser elegido, y su elección es independiente de las demás extracciones.

Se piensa que las personas son generadores aleatorios imperfectos, hay estudios que demuestran que existen tendencias claras en los humanos para la elaboración de ***secuencias sesgadas*** y están relacionadas con características personales, con los conocimientos o informaciones previas o con la edad.

Métodos manuales: *lanzamiento de monedas, lanzamientos de dados, dispositivos mecánicos, dispositivos electrónicos.*
Métodos de computación analógica: *son métodos que dependen de ciertos procesos físicos aleatorios, por ejemplo, el comportamiento de una corriente eléctrica.*

Tablas de bibliotecas: *son números aleatorios que se han publicado; de los cuales podemos encontrar listas en los libros de probabilidad y tablas de matemáticas. Estos números fueron generados por alguno de los métodos de computación analógica.*
Métodos de computación digital: *cuando se usa una computadora digital.*

Números pseudoaleatorios

Son unos números generados por medio de una ***función*** *(determinista, no aleatoria)* y que *aparentan* ser aleatorios. Estos números pseudoaleatorios se generan a partir de un **valor inicial** aplicando iterativamente la función. La sucesión de números pseudoaleatorios es sometida a diversos tests para medir hasta qué punto se asemeja a una sucesión aleatoria .

¿Por qué hay que recurrir a los número pseudoaleatorios?
Fundamentalmente porque las sucesiones de números pseudoaleatorios son más rápidas de generar que las de números aleatorios. Si las personas tenemos dificultad en generar números aleatorios, mucho más la tiene una computadora, la dificultad está en que una computadora es tan "torpe" que no sabe generarlos *(es solo una metáfora)*. Por eso usan números pseudoaleatorios, que para nuestro fin es lo mismo, pues nadie los puede predecir.

Método del cuadrado medio

Ante las enormes posibilidades científicas que se vislumbraban ante el creciente uso de las primeras grandes computadoras electromecánicas *(MARK I, ENIAC, UNIVAC, etc.)*, las cuales mediante el uso de tarjetas perforadas o cintas magnéticas permitían ejecutar diversas instrucciones digitales para realizar operaciones muy complejas, entró en escena el matemático ***John von Neumann (1903–1957)***, quien hacia 1946 propuso usar el algoritmo conocido como el **"Método del Cuadrado Medio"** para la generación matemática de números pseudoaleatorios.


MARK I


ENIAC


UNIVAC

El Método del Cuadrado Medio de von Neumann consistía en el uso de un algoritmo en el cual al inicio se introduce un número cualquiera conformado por 10 dígitos, luego se calcula el cuadrado de ese número inicial, a continuación se toman exactamente los 10 dígitos ubicados en la mitad del número resultante, y ese número conformado por los 10 dígitos se toma como un nuevo número aleatorio que sirve para engrosar la secuencia aleatoria generada, al cual posteriormente se le puede aplicar de nuevo el algoritmo del cuadrado medio para así obtener sucesivamente más números aleatorios.

El esquema de funcionamiento del algoritmo del Método del Cuadrado Medio propuesto por John von Neumann es el siguiente:

MÉTODO DEL CUADRADO MEDIO PARA GENERAR NÚMEROS ALEATORIOS:			
No. de arranque:	Resultado al ser elevado al cuadrado (x^2):	Selección de los 10 dígitos del medio:	Nuevo número aleatorio generado:
5772156649	33317792380594909201	33317792380594909201	7923805949
7923805949	62786700717407800000	62786700717407800000	7007174078
7007174078	49100488559395200000	49100488559395200000	4885593952
4885593952	23869028263819000000	23869028263819000000	0282638190

En esta tabla se observa que cada nuevo número aleatorio que al final es generado por la aplicación del algoritmo del Cuadrado Medio puede luego ser tomado como un nuevo número de arranque (o "Número Semilla") para volver a generar un nuevo número aleatorio, proceso que se supone es perpetuo hasta el infinito y con resultados siempre impredecibles.

John von Neumann tenía la gran esperanza de que con la ayuda de las nuevas computadoras que se estaban construyendo se podrían diseñar y usar algoritmos cada vez más complejos, que servirían para producir listados de números aleatorios que al ser sometidos a los Tests Estadísticos más comunes no reflejarían la presencia de la más mínima *tendencia*, *desviación*, *regresión* o *repetición periódica*.
Sin embargo, el mismo ***Método del Cuadrado Medio***. más tarde reflejó que en cierto momento la secuencia de los números aleatorios generados puede caer en un **"loop"**, es decir, autónomamente cae en un ciclo repetitivo de resultados que se denomina **"periodo"**, lo cual le resta el pretendido carácter aleatorio e impredecible a los números generados por ese método.

*Esto ocurre principalmente cuando en cierto momento el número **cero (0)** comienza a aparecer muchas veces dentro de los **10 dígitos** que conforman el **supuesto número aleatorio** resultante de la elevación al*

cuadrado, caso en el cual cuando esos ceros son elevados de nuevo al cuadrado seguirán repitiéndose en mayor medida y por siempre dentro de la secuencia de los nuevos números generados.
*Esto lo demostró **G. E. Forsythe** en los años 50's al aplicar el Método del Cuadrado Medio a números que dentro de los 10 dígitos incluían bastantes ceros, con lo que obtuvo secuencias repetitivas de números generados que al poco tiempo **se hacían fácilmente predecibles**.*

Otros matemáticos probaron el **Método del Cuadrado Medio** usando números de hasta 20 dígitos, o incluso usando números binarios, y se logró establecer que **existen 13 posibles ciclos o periodos repetitivos** diferentes en los que pueden caer las secuencias de los números generados por este método, y además **se probó que cada uno de estos periodos puede comenzar a repetirse de manera variable**, es decir, bien puede comenzar a repetirse cada 150 resultados aparecidos, o bien puede comenzar a repetirse cada 2.000 resultados generados, o bien cada 12.345 resultados, etc., todo lo cual depende de los ***"Números–Semilla"*** que inicialmente sean introducidos como arranque para dar inicio a este algoritmo.
En otras palabras, si los números generados por el Método del Cuadrado Medio terminan reflejando una secuencia repetitiva o un ciclo que ocurre periódicamente cada tantos resultados aparecidos *(un "loop")*, entonces esos números no deberían ser considerados completamente aleatorios sino ***sólo pseudoaleatorios***.

$$R_{n+1} = \text{mid}(R_n^2, m)$$

where:
R_{n+1} = *new random number*
R_n = *previous random number*
R_0 = *initial seed value*
mid(,m) = *m number of digits extracted from the relative middle*

Generadores de números aleatorios

Un paso clave en simulación es tener algoritmos que generen variables aleatorias con distribuciones específicas, como la exponencial, normal, etc.
Esto es efectuado en dos fases:

- ***La primera consiste en generar una secuencia de números aleatorios distribuidos "uniformemente" entre 0 y 1.***
- ***Luego esta secuencia es transformada para obtener los valores aleatorios de las distribuciones deseadas.***

Las propiedades deseadas del generador son las siguientes:

1. **Deben ser eficientes computacionalmente**: dado que típicamente se requieren varios miles de números aleatorios por corrida, el tiempo de procesador requerido para generarlos debe ser pequeño.

2. **El periodo debe ser largo**: periodos cortos limitan la longitud aprovechable de una corrida de simulación porque el reciclaje resulta en una repetición de secuencias de eventos.

3. <u>Los valores sucesivos deben ser independientes y uniformemente distribuidos</u>: la correlación entre números sucesivos debe ser pequeña y si es significante indica dependencia.

Generadores Congruenciales Lineales (GCL)

Los principales generadores de números pseudoaleatorios utilizados en la actualidad son los llamados generadores ***congruenciales lineales***, introducidos por **Lehmer en 1951**. Un método congruencial comienza con un valor inicial *(semilla)* x_0, y los sucesivos valores x_n, para $n \geq 1$, se obtienen con la siguiente fórmula:

$$x_n = ax_{n-1} + b \; mod \; m$$

Donde $\boldsymbol{a}, \boldsymbol{m}, \; \boldsymbol{b}$ son enteros positivos que se denominan, respectivamente, el **multiplicador**, el **módulo** y el **incremento**. Si $b = 0$ el generador se denomina multiplicativo; en caso contrario se denomina mixto. La selección de $a, m, \; b$ afectan el periodo y la autocorrelación en la secuencia.

Si m es primo, distintas elecciones de a, permiten generar un periodo completo de (m-1) valores.

Entre los resultados de los estudios realizados con estos generadores tenemos:
1. El módulo m debe ser grande. Dado que los x están entre 0 y m-1, el periodo nunca puede ser mayor que m.
2. Para que el computo de mod m sea eficiente, m debe ser una potencia de 2, es decir, $\boldsymbol{2^k}$. En este caso mod m puede ser obtenido truncando el resultado y tomando en k bits a la derecha.
3. Si b es diferente de cero, el periodo máximo posible m se obtiene si y sólo si:

a) Los enteros m y b son primos relativos -- no tengan factores comunes excepto el 1.
b) Todo número primo que sea un factor de m lo es también de a-1.
c) a-1 es un múltiplo de 4 si m es un múltiplo de 4.

Todas estas condiciones se cumplen si $m = 2^k$, a = 4c + 1, y b es impar, donde c, b, y k son enteros positivos.
Si un generador tiene el periodo máximo posible se llama generador de **periodo completo**.
Todos los generadores de periodo completo no son igualmente buenos. Son preferibles los generadores con **menor autocorrelación** entre números sucesivos.
Por ejemplo, los dos generadores siguientes son de periodo completo, pero el primero tiene una correlación de 0.25 entre x_{n-1} y x_n , mientras que el segundo tiene una correlación despreciable de menos de 2^{-18}.

$$x_n = \left((2^{34} + 1)x_{n-1} + 1\right) mod \; 2^{35}$$
$$x_n = \left((2^{18} + 1)x_{n-1} + 1\right) mod \; 2^{35}$$

```
programa prueba_random;
        variables
                i,x: entero;
                ran: real;
```

```
funcion random(var x:entero): real;
        constantes
                a = 16807; # multiplicador
                m = 2147483647; # modulo
                q = 127773; {#m div a
                r = 2836; # m mod a
        comienzo
                x = a*(x mod q) - r*(x div q);
                si (x < 0) hacer x = x + m;
                random = x/m;
        fin;
comienzo
        x:=1; #semilla
        para i=1 a 10000 hacer ran=random(x);
        escribir(x);
        #Salida = 1043618065
fin.
```

Generadores de Fibonacci extendidos.

Una secuencia de Fibonacci $\{x_n\}$ se genera por la siguiente relación:

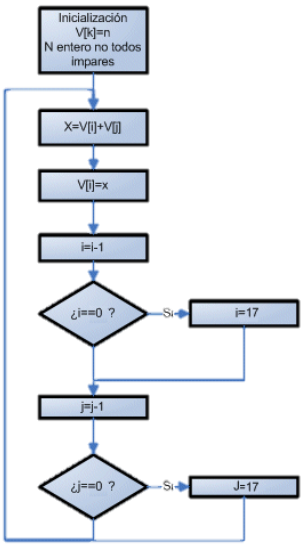
$$x_n = x_{n-1} + x_{n-2}$$

Se puede intentar usar un generador de la forma:

$$\boldsymbol{x_n = (x_{n-1} + x_{n-2}) mod\ n}$$

Sin embargo esta secuencia no tiene buenas propiedades aleatorias y en particular tiene ***alta correlación serial***. El siguiente generador, que sigue este enfoque, pasa la mayoría de las pruebas estadísticas:

$$\boldsymbol{x_n = (x_{n-5} + x_{n-17}) mod\ 2^k}$$



Para implementar este generador se pueden usar 17 localidades de memoria L[1], ..., L[17] las cuales son inicializadas con 17 enteros que no sean todos pares.
Fijamos i y j en 5 y 17 respectivamente, y el siguiente procedimiento es ejecutado para obtener los números aleatorios:

x= L[i] + L[j];
L[i]= x;
i= i - 1; si i=0 entonces i= 17;
j= j - 1; si j=0 entonces j= 17;

La adición en la primera línea es automáticamente $\boldsymbol{mod\ 2^k}$ en máquinas de k-bits y Aritmética de complemento a 2.
El periodo del generador es $\boldsymbol{2^k\left(2^{17}-1\right)}$ que es considerablemente mayor al que se puede obtener con GCL.

Generadores Combinados.

Es posible combinar generadores para obtener "mejores" generadores. Algunas de las técnicas usadas son:

1. **OR-exclusivo** de números aleatorios de dos o más generadores. Esta técnica es similar a la anterior excepto que la suma es reemplazada por un *or-exclusivo* bit por bit. Se ha demostrado que esta técnica aplicada a números ligeramente aleatorios puede ser usada para generar números con mayor aleatoriedad.
2. **Barajeo**. Usa una secuencia como un índice para decidir qué número generado por otra secuencia será retornado. Por ejemplo, uno de estos algoritmos usa un arreglo de tamaño 100 que contiene números de una secuencia aleatoria xn. Para generar un número aleatorio se genera un número aleatorio yn (entre 0 y m-1) para obtener el índice i = 1 + 99yn / (m-1). El valor del i-esimo elemento del arreglo es devuelto. Un nuevo valor xn es calculado y almacenado en la i-esima localidad.

Aunque estos generadores caen dentro de la sección anterior, su relevancia amerita su discusión por separado.
Consideremos el generador $x_n = 7^5 x_{n-1}\ mod(2^{31} - 1)$
En la medida que las computadoras se han vuelto más rápidas, la longitud de su ciclo se ha tornado inadecuada ya que se corre el riesgo de que, en unas cuantas horas de simulación, la secuencia se agote y se repita varias veces, trayendo como consecuencia serias dudas en cuanto a la validez de los resultados.

Supongamos que tenemos una máquina con un procesador con 109 ciclos o tics del reloj por segundo. Supongamos también que el procesador es capaz de generar un número aleatorio por *ciclo (esto es muy optimista ya que en realidad se requieren varios ciclos para producir un número aleatorio debido a las operaciones involucradas)*.

Bajo estas condiciones se agotaría la secuencia en $2{,}1\ .\frac{10^9}{10^9} = 2.1$ seg
Por supuestos que será en más tiempo, pero se observa que efectivamente esta secuencia es fácilmente agotable durante una simulación.
Una manera para conseguir generadores con periodos más largos consiste en sumar números aleatorios de dos o más generadores.

Generadores Normalmente Usados.

Un generador GCL muy popular y que ya hemos mencionado es:

$$x_n = 7^5 x_{n-1}\ mod(2^{31} - 1)$$

Éste es usado en el sistema SIMPL/I de IBM (1972), APL de IBM (1971), el sistema operativo PRIMOS de Prime Computer (1984), y la librería científica de IMSL (1987). Tiene buenas propiedades aleatorias y es recomendado como un estándar mínimo.

Un estudio de generadores multiplicativos GCL con módulo $m = 2^{31} - 1$ que comparó su eficiencia y aleatoriedad, recomienda los dos siguientes como los mejores:

$$x_n = 48271 x_{n-1}\ mod(2^{31} - 1)$$
$$x_n = 69621 x_{n-1}\ mod(2^{31} - 1)$$

El siguiente generador es usado en SIMSCRIPT II.5 y en FORTRAN DEC-20:

$$x_n = 6630360016x_{n-1}\ mod(2^{31} - 1)$$

El siguiente generador es usado en el sistema Pascal de la Universidad de Sheffield para computadores Prime:

$$x_n = 16807x_{n-1}\ mod\ 2^{31}$$

Dado que 16.807 no es de la forma 8i ± 3, este generador no tiene el periodo máximo posible de 2^{31-2}. También es usado en la subrutina UNIFORM del paquete estadístico SAS, pero una técnica de barajeo es usada para mejorar la aleatoriedad.

SIMULA en UNIVAC usa:

$$x_n = 5^{13}x_{n-1}2^{35}$$

Algunos autores dicen que no tiene buenas propiedades aleatorias.

El sistema operativo UNIX soporta en siguiente GCL mixto:

$$x_n = (1103515245x_{n-1} + 12345)mod\ 2^{32}$$

***NOTA**: Diseñar nuevos generadores parece muy simple, pero muchos generadores propuestos por expertos estadísticos fueron encontrados deficientes. Por lo tanto es mejor usar un generador que ha sido extensamente probado en vez de inventar uno nuevo.*

Selección de Semilla.

En principio la semilla no debería afectar los resultados de la simulación. Sin embargo, una mala combinación de semilla y generador pueden producir conclusiones erróneas.
Si el generador es de periodo completo y solo se requiere una variable aleatoria, cualquier semilla es buena. Hay que tener especial cuidado en simulaciones que requieren números aleatorios para más de una variable (simulaciones de secuencias múltiples), que es la mayoría de los casos. Por ejemplo, la simulación de una cola simple requiere generar llegadas y servicios aleatorios y requiere dos secuencias de números aleatorios.
1. **No use cero**. Cero funciona para generadores GCL mixtos pero hace que los multiplicativos se queden en cero.
2. **Evite valores pares**. Si un generador no es de periodo completo (por ejemplo GCL multiplicativo con modulo $\mathrm{m} = 2^k$) la semilla debe ser impar. En otros casos no importa.
3. **No subdivida una secuencia**. Usar una única secuencia para todas las variables es un error común. Por ejemplo, en la secuencia {u1 , u2 , ...} generada a partir de la semilla u0 , el analista usa u1 para generar el tiempo entre llegadas, u2 para el tiempo de servicio, etc. Esto puede resultar en una fuerte correlación entre las variables.

4. **Use secuencias que no se solapan**. Cada secuencia requiere su semilla. Si la semilla es tal que hace que dos secuencias se solapen, habrá correlación entre las secuencias y estas no serán independientes.

Consideremos el ejemplo trivial de iniciar las dos secuencias de una cola simple con la misma semilla, lo cual haría las secuencias idénticas e introduciría una fuerte correlación positiva entre los tiempos entre llegadas y los tiempos de servicio. Esto puede llevar a conclusiones erróneas.
Hay que seleccionar las semillas de forma tal que las secuencias no se solapen. Si {u1 , u2 , ...} generada a partir de la semilla u0 , y necesitamos por ejemplo 10.000 tiempos entre llegadas, 10.000 tiempos de servicios, etc., podemos seleccionar u0 como la semilla de la primera secuencia, u10.000 para la segunda, u20.000 para la tercera, etc. Los un se pueden determinar mediante un programa de prueba que llama al generador o se pueden calcular directamente para generadores GCL mixtos o multiplicativos (b = 0) con la formula siempre que los cálculos sean exactos:

$$x_n = a^n x_0 + \frac{b(a^n - 1)}{a - 1} mod\ m$$

5. **Reuse semillas en replicaciones sucesivas**. Si el experimento es replicado varias veces, la secuencia no necesita ser reinicializada y se puede usar la semilla dejada en la replicación previa.
6. **No use semillas aleatorias**. Semillas aleatorias, como por ejemplo la hora del día, causan dos problemas:
   - La simulación no puede ser reproducida.
   - No se puede garantizar que secuencias múltiples no se solapen.

Números aleatorios con distribución no uniforme.

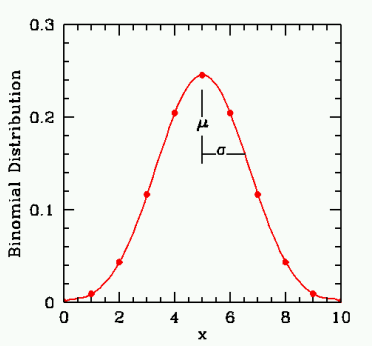
Una variable aleatoria es una función que asume sus valores de acuerdo a los resultados de un experimento aleatorio, es decir, un experimento donde existe incertidumbre acerca del resultado que va a ocurrir.

Una ***variable aleatoria es discreta*** si su rango de valores es un conjunto finito o infinito enumerable.
Existen una infinidad de variables aleatorias discretas, entre las más conocidas están: ***la Binomial, la geométrica, la hipergeométrica, la Poisson y la Binomial Negativa***.
Si la variable aleatoria discreta X tiene rango de valores RX entonces la función **p(k)=Prob[X=k]** donde **x∈RX** es llamada la función de probabilidad de **X**.
Asimismo, la función $F(t) = P[X \leq t] = \sum_{k \leq t} P[X = k]$ es llamada la función de distribución acumulativa de X.
Una ***variable aleatoria continua*** es aquella cuyo rango de valores es cualquier intervalo de la recta real, entre las más conocidas están: ***la uniforme, la exponencial, la gamma, la Ji-Cuadrado, la Beta, la Normal, la t de Student, la Cauchy, la Weibull,etc***.
Si la variable aleatoria continua X tiene rango de valores RX entonces existe una función no-negativa $f(x)$ tal que $P(a < X < b) = \int_a^b f(x)dx$
Asimismo, la función

$$F(t) = P[X \leq t] = \int_{-\infty}^{t} f(x)dx$$

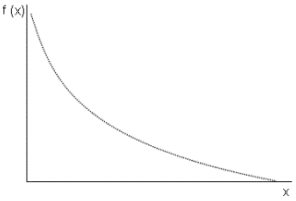
es llamada la función de distribución acumulativa de X.
En los modelos estocásticos existirán una o más variable aleatorias interactuando.
Estas variables siguen distribuciones de probabilidad teóricas o empíricas, diferentes a la distribución uniforme (0-1).

Binomial: Es una distribución de probabilidad discreta que cuenta el número de éxitos en una secuencia de n ensayos de Bernoulli independientes entre sí, con una probabilidad fija p de ocurrencia del éxito entre los ensayos. Un experimento de Bernoulli se caracteriza por ser dicotómico, esto es, sólo son posibles dos resultados. A uno de estos se denomina éxito y tiene una probabilidad de ocurrencia p y al otro, fracaso, con una probabilidad q = 1 - p. En la distribución binomial el anterior experimento se repite n veces, de forma independiente, y se trata de calcular la probabilidad de un determinado número de éxitos. Para n = 1, la binomial se convierte, de hecho, en una distribución de Bernoulli.

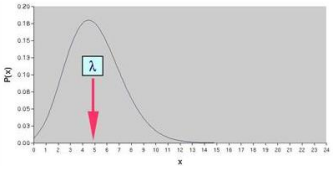


Geométrica: Es cualquiera de las dos distribuciones de probabilidad discretas siguientes:
- la distribución de probabilidad del número X del ensayo de Bernoulli necesaria para obtener un éxito, contenido en el conjunto { 1, 2, 3,...} o
- la distribución de probabilidad del número Y = X − 1 de fallos antes del primer éxito, contenido en el conjunto { 0, 1, 2, 3,... }.

Cual de éstas es la que uno llama "la" distribución geométrica, es una cuestión de convención y conveniencia.



Poisson: es una distribución de probabilidad discreta que expresa, a partir de una frecuencia de ocurrencia media, la probabilidad de que ocurra un determinado número de eventos durante cierto período de tiempo. Concretamente, se especializa en la probabilidad de ocurrencia de sucesos con probabilidades muy pequeñas, o sucesos "raros".



Para generar números que sigan el comportamiento de estas variables, se pueden utilizar algunos métodos como los siguientes:

1. Método de la transformada inversa
2. Método de rechazo
3. Método de composición
4. Procedimientos especiales

Método de la transformada inversa: El método de la transformada inversa utiliza la distribución acumulada F(x) de la distribución que se va a simular.
Puesto que F(x) está definida en el intervalo (0-1), se puede generar un número aleatorio uniforme R y tratar de determinar el valor de la variable aleatoria para cual su distribución acumulada es igual a R, es decir, el valor simulado de la variable aleatoria que sigue un distribución de probabilidad f(x), se determina al resolver la siguiente ecuación.

$$F(x) = R \text{ ó } x = F\text{^}{-1}(R)$$

La dificultad principal de este método descansa en el hecho de que en algunas ocasiones es difícil encontrar la transformada inversa. Sin embargo, si esta función inversa ya ha sido establecida, generando

números aleatorios uniformes se podrán obtener valores de la variable aleatoria que sigan la distribución de probabilidad deseada.

Método de aceptación y rechazo: La idea aquí es que se tiene una variable aleatoria Y con función de densidad g(y), la cual puede ser generada fácilmente.
Se desea generar otra variable X con función de densidad f(x), para ello se genera un valor de Y con densidad g(y) y se toma x=Y con probabilidad proporcional a f(y)/g(y).
Esto es: $\mathrm{P}[X = y] = k\, f(y)/g(y)$ donde k es la constante de proporcionalidad.
Si c es una constante tal que $f(y)/g(y) \leq c$ para todo y
Entonces el siguiente sería el algoritmo de aceptación y rechazo para generar una variable aleatoria X con función de densidad f(x).

Paso 1: Generar Y con densidad g
Paso 2: Generar una variable aleatoria uniforme U(0,1)
Paso 3: Si $\mathrm{U} \leq f(y)/cg(y)$ entonces X=Y de lo contrario volver al paso 1

Mientras más cerca se encuentre f de g más rápidamente se obtendrá la cantidad deseada de valores aleatorios de X.
Usualmente se toma la densidad uniforme como la densidad g(y) y en ese caso el método de aceptación y rechazo es llamado “hit and miss”.

Método de composición-descomposición: Se basa en la idea de dividir la f(x) original en una combinación de fi (x) cuya selección se hace en
base a minimizar el tiempo de computación requerido.
Algoritmo:

* Dividir la f(x) original en sub-áreas.
* Definir una distribución de probabilidad para cada sub-área:
* Expresar la distribución original como:
* Obtener la distribución acumulada de las áreas
* Generar dos números aleatorios uniformemente distribuidos, R1 y R2.
*Con R1 entrar a la distribución acumulada de las áreas (por el eje y) y seleccionar cuál fi (x) se va a usar (método de la transformada inversa).

· Utilizar R2 para simular x por el método de la transformada inversa con fi (x).
· Repetir generando nuevos pares de números aleatorios.

Generación de permutaciones aleatorias.

Pensando en un juego de azar, por ejemplo un juego de cartas; uno de los jugadores tiene el objetivo de mezclar el mazo, el resto de los jugadores “verifican” que no pueda realizar ninguna acción deshonesta. Pero, ¿qué pasa con la llegada de Internet y los juegos que se realizan en la red, por ejemplo los juegos de póker?
En este tipo de juego, los jugadores no pueden realizar esta acción de “verificar” de la misma manera que en un juego presencial.
Pues bien, esa “verificación” puede ser garantizada por dos caminos:

• Por quien cubre el rol de casino on-line y todos los jugadores aceptan esa posición y entienden que es una posición de privilegio.

• Por un protocolo de computación criptográfico multiparte, el cual asegura que ningún jugador tendrá alguna ventaja con el resto, considerando que cada participante “conocerá” sólo una parte del “secreto” que equivale al reparto de cartas.

Dos consideraciones debemos hacer en este esquema:

a- Existe un grupo de participantes **P** = P1,…….,Pn

b- Existe un secreto **s**.